ENV 790.30 - Time Series Analysis for Energy Data | Spring 2022

Assignment 5 - Due date 02/28/22

Tatiana Sokolova

Directions

You should open the .rmd file corresponding to this assignment on RStudio. The file is available on our class repository on Github. And to do so you will need to fork our repository and link it to your RStudio.

Once you have the project open the first thing you will do is change "Student Name" on line 3 with your name. Then you will start working through the assignment by **creating code and output** that answer each question. Be sure to use this assignment document. Your report should contain the answer to each question and any plots/tables you obtained (when applicable).

When you have completed the assignment, **Knit** the text and code into a single PDF file. Rename the pdf file such that it includes your first and last name (e.g., "LuanaLima_TSA_A05_Sp22.Rmd"). Submit this pdf using Sakai.

R packages needed for this assignment are listed below. Install these packages, if you haven't done yet. Do not forget to load them before running your script, since they are NOT default packages.\

```
#Load/install required package here
library(xlsx)
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(tseries)
library(ggplot2)
library(Kendall)
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(tidyverse)  #load this package so yon clean the data frame using pipes
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.6     v dplyr   1.0.7
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
## v purrr   0.3.4
```

```
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x lubridate::as.difftime() masks base::as.difftime()
## x lubridate::date()        masks base::date()
## x dplyr::filter()          masks stats::filter()
## x lubridate::intersect()   masks base::intersect()
## x dplyr::lag()             masks stats::lag()
## x lubridate::setdiff()     masks base::setdiff()
## x lubridate::union()       masks base::union()
```

Decomposing Time Series

Consider the same data you used for A04 from the spreadsheet "Table_10.1_Renewable_Energy_Production_and_Consump
The data comes from the US Energy Information and Administration and corresponds to the January 2021
Monthly Energy Review.

```
#Importing data set - using xlsx package
energy_data <- read.xlsx(file="../Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Source

#Now let's extract the column names from row 11 only
read_col_names <- read.xlsx(file="../Data/Table_10.1_Renewable_Energy_Production_and_Consumption_by_Sou

colnames(energy_data) <- read_col_names
head(energy_data)
```

```
##        Month Wood Energy Production Biofuels Production
## 1 1973-01-01                129.630       Not Available
## 2 1973-02-01                117.194       Not Available
## 3 1973-03-01                129.763       Not Available
## 4 1973-04-01                125.462       Not Available
## 5 1973-05-01                129.624       Not Available
## 6 1973-06-01                125.435       Not Available
##   Total Biomass Energy Production Total Renewable Energy Production
## 1                         129.787                             403.981
## 2                         117.338                             360.900
## 3                         129.938                             400.161
## 4                         125.636                             380.470
## 5                         129.834                             392.141
## 6                         125.611                             377.232
##   Hydroelectric Power Consumption Geothermal Energy Consumption
## 1                         272.703                         1.491
## 2                         242.199                         1.363
## 3                         268.810                         1.412
## 4                         253.185                         1.649
## 5                         260.770                         1.537
## 6                         249.859                         1.763
##   Solar Energy Consumption Wind Energy Consumption Wood Energy Consumption
## 1            Not Available          Not Available                 129.630
```

```
## 2            Not Available           Not Available                 117.194
## 3            Not Available           Not Available                 129.763
## 4            Not Available           Not Available                 125.462
## 5            Not Available           Not Available                 129.624
## 6            Not Available           Not Available                 125.435
##   Waste Energy Consumption Biofuels Consumption
## 1                    0.157        Not Available
## 2                    0.144        Not Available
## 3                    0.176        Not Available
## 4                    0.174        Not Available
## 5                    0.210        Not Available
## 6                    0.176        Not Available
##   Total Biomass Energy Consumption Total Renewable Energy Consumption
## 1                          129.787                             403.981
## 2                          117.338                             360.900
## 3                          129.938                             400.161
## 4                          125.636                             380.470
## 5                          129.834                             392.141
## 6                          125.611                             377.232
```

```
nobs=nrow(energy_data)
nvar=ncol(energy_data)
```

Q1

For this assignment you will work only with the following columns: Solar Energy Consumption and Wind Energy Consumption. Create a data frame structure with these two time series only and the Date column. Drop the rows with *Not Available* and convert the columns to numeric. You can use filtering to eliminate the initial rows or convert to numeric and then use the drop_na() function. If you are familiar with pipes for data wrangling, try using it!

```
#creating df structure for columns of interest

energy_data$Month = as.Date(energy_data$Month, format = "%m/%d/%Y")
energy_data$'Solar Energy Consumption' <- as.numeric(energy_data$'Solar Energy Consumption')
```

```
## Warning: NAs introduced by coercion
```

```
energy_data$'Wind Energy Consumption' <- as.numeric(energy_data$'Wind Energy Consumption')
```

```
## Warning: NAs introduced by coercion
```

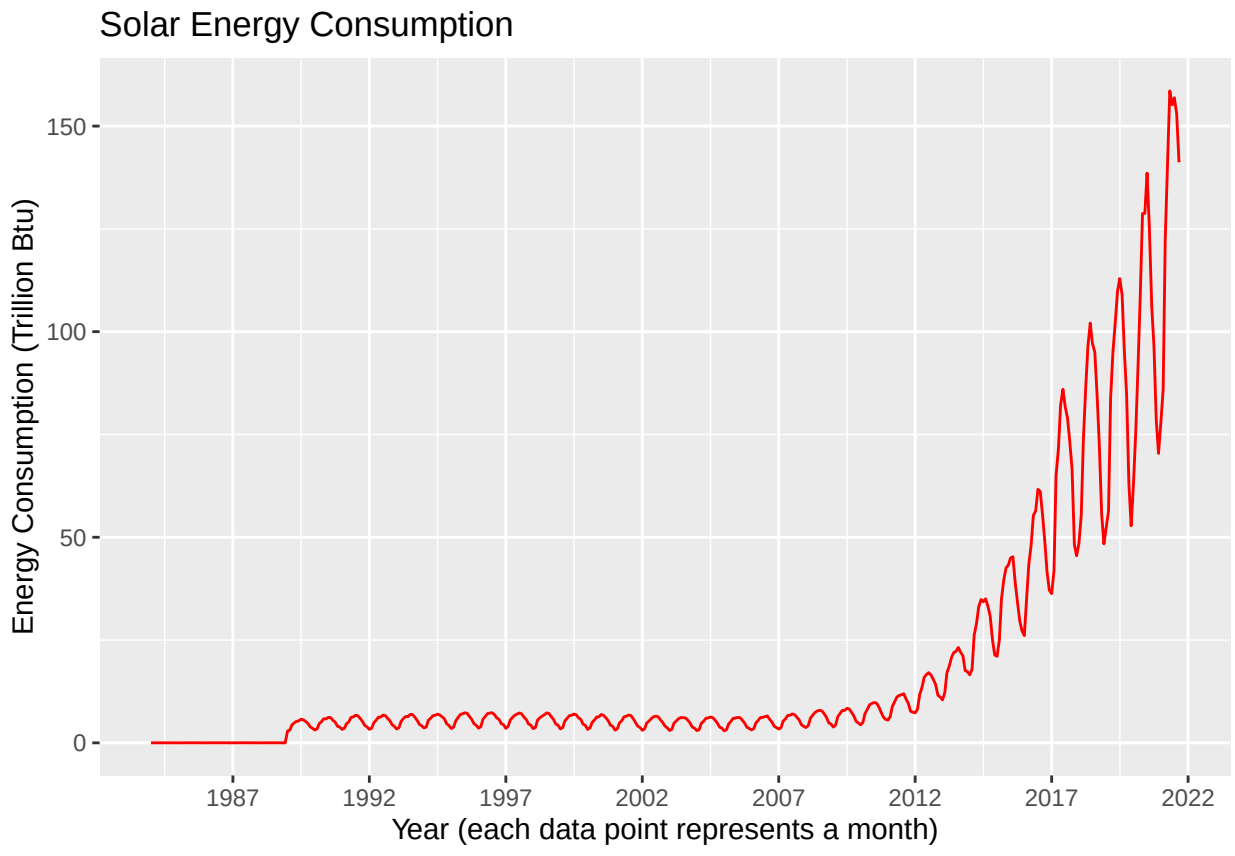
```
df <- energy_data[,c('Month','Solar Energy Consumption','Wind Energy Consumption')]

df_clean <- df %>%
            drop_na('Solar Energy Consumption') %>%
            drop_na('Wind Energy Consumption')

head(df_clean) #confirming nas removed
```

```
##        Month Solar Energy Consumption Wind Energy Consumption
## 1 1984-01-01                  -0.001                   0.000
## 2 1984-02-01                   0.001                   0.002
## 3 1984-03-01                   0.002                   0.002
## 4 1984-04-01                   0.003                   0.006
## 5 1984-05-01                   0.007                   0.008
## 6 1984-06-01                   0.010                   0.006
```

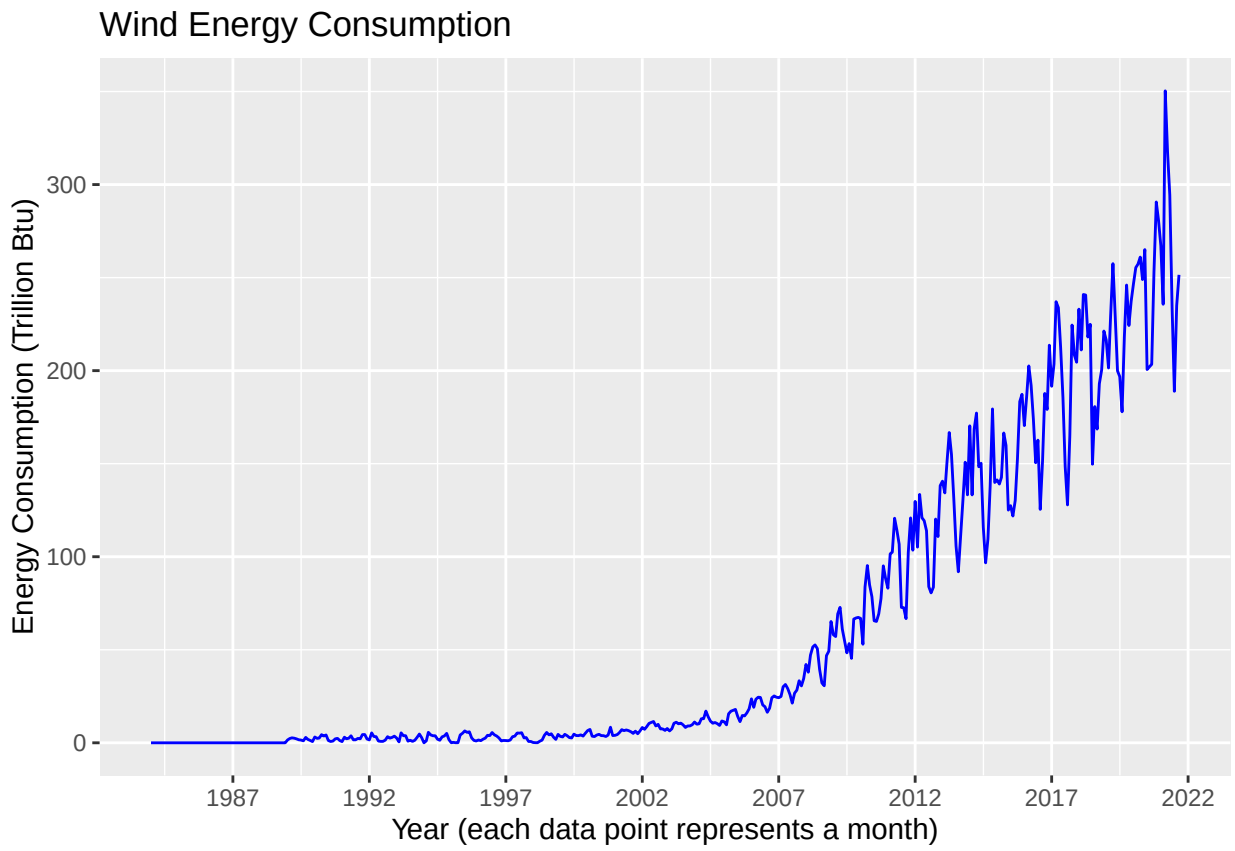
Q2

Plot the Solar and Wind energy consumption over time using ggplot. Plot each series on a separate graph. No need to add legend. Add informative names to the y axis using `ylab()`. Explore the function scale_x_date() on ggplot and see if you can change the x axis to improve your plot. Hint: use *scale_x_date(date_breaks = "5 years", date_labels = "%Y")"*

```
#solar only
ggplot(df_clean, ) +
  geom_line(aes(x=df_clean[,1], y= df_clean[,2]), color = "red") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")+
  ggtitle("Solar Energy Consumption")+
  ylab(paste0("Energy Consumption (Trillion Btu)")) +
  xlab(paste0("Year (each data point represents a month)"))
```



```
#wind only
ggplot(df_clean, ) +
  geom_line(aes(x=df_clean[,1], y= df_clean[,3]), color = "blue") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")+
  ggtitle("Wind Energy Consumption")+
  ylab(paste0("Energy Consumption (Trillion Btu)")) +
  xlab(paste0("Year (each data point represents a month)"))
```

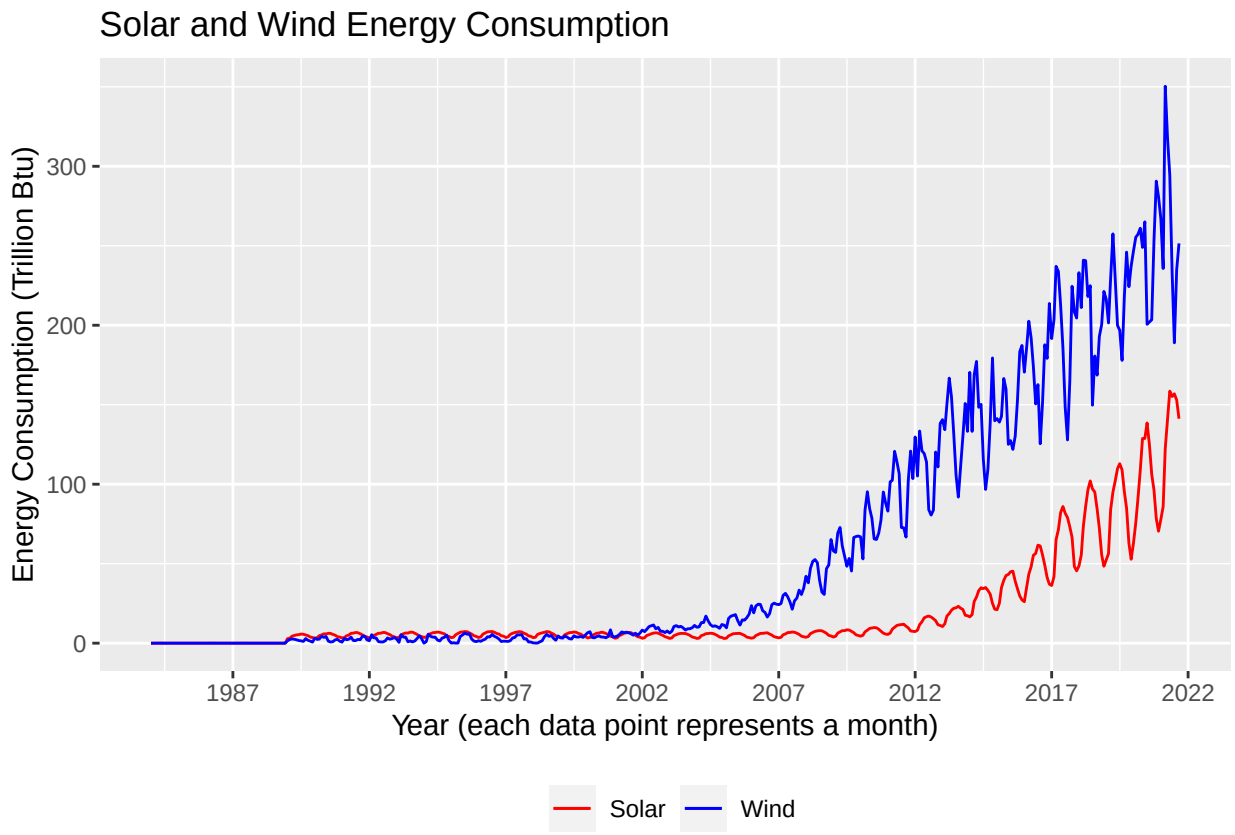


Q3

Now plot both series in the same graph, also using ggplot(). Look at lines 142-149 of the file `05_Lab_OutliersMissingData_Solution` to learn how to manually add a legend to ggplot. Make the solar energy consumption red and wind energy consumption blue. Add informative name to the y axis using `ylab("Energy Consumption)`. And use function scale_x_date() again to improve x axis.

```
ggplot(df_clean, aes(x=df_clean[,1], y=df_clean[,2])) +
  geom_line(aes(x=df_clean[,1], y= df_clean[,2], color = "Solar")) +
  geom_line(aes(x=df_clean[,1], y = df_clean[,3], color = "Wind")) +
  labs(color="") +
  scale_x_date(date_breaks = "5 years", date_labels = "%Y")+
  scale_color_manual(values = c("Solar" = "red","Wind" = "blue"),
                                labels=c("Solar", "Wind")) +
  theme(legend.position = "bottom") +
  ggtitle("Solar and Wind Energy Consumption")+
```

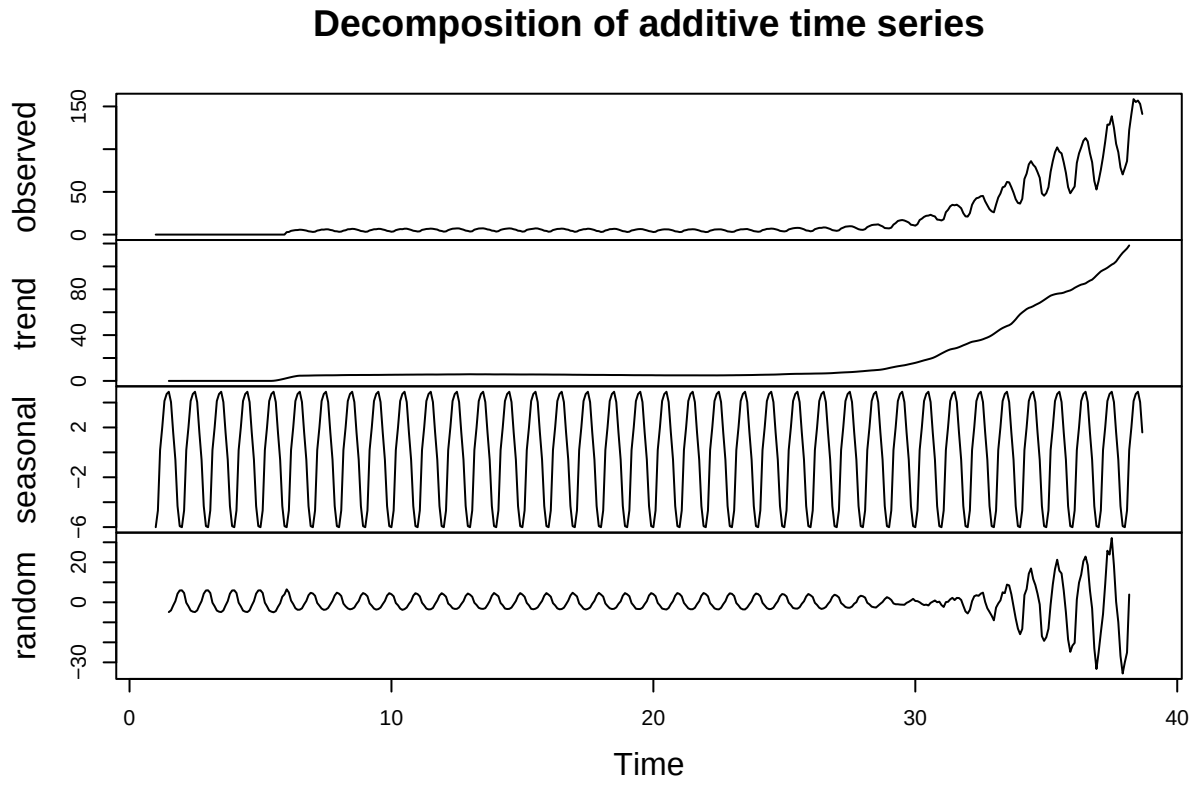
```
  ylab(paste0("Energy Consumption (Trillion Btu)")) +
  xlab(paste0("Year (each data point represents a month)"))
```



Q3

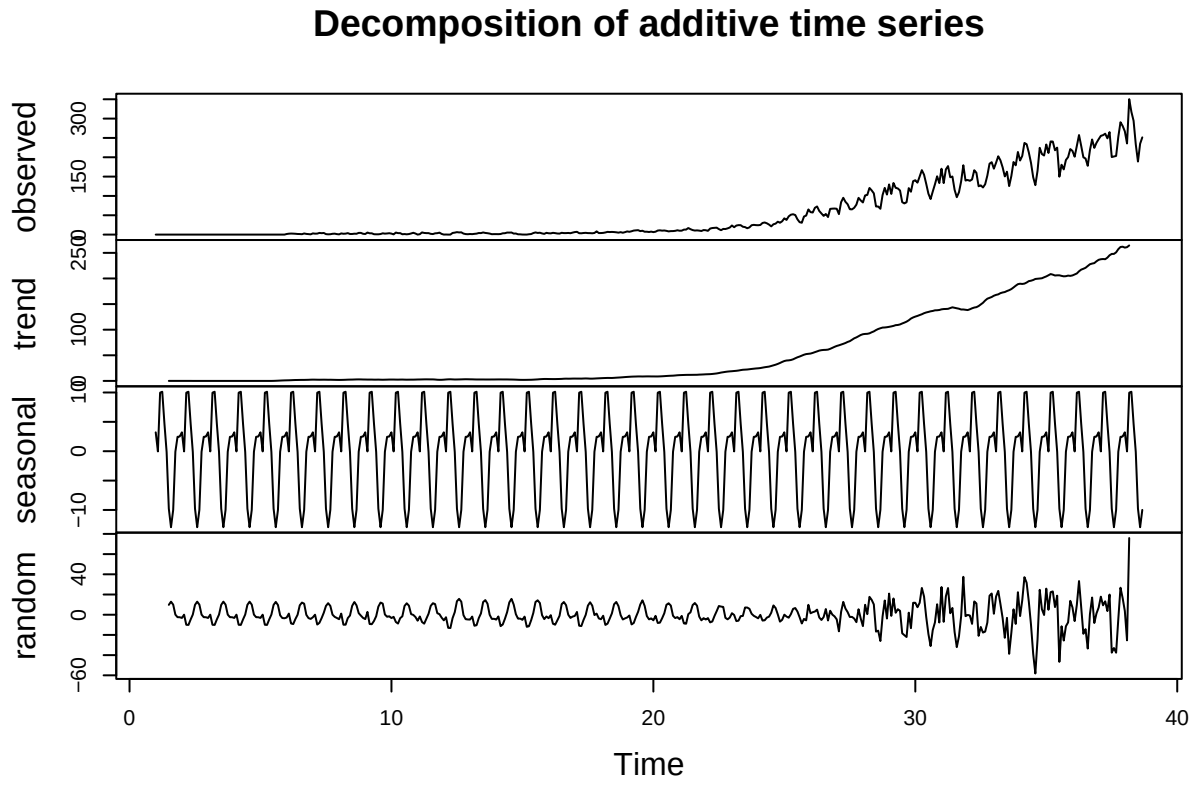
Transform wind and solar series into a time series object and apply the decompose function on them using the additive option, i.e., `decompose(ts_data, type = "additive")`. What can you say about the trend component? What about the random component? Does the random component look random? Or does it appear to still have some seasonality on it?

```
#solar series
ts_solar_data <- ts(df_clean[,2],frequency=12)

#decomposition
decompose_solar_data=decompose(ts_solar_data, type="additive")
plot(decompose_solar_data)
```



Observations: The trend component identifies a clearly increasing trend of solar energy consumption. The random component shows minor variability throughout the dataset until the end of the series where there is an increase in variability. The random component does not look very random and appears to have its own seasonal trend.

```
#wind series ts transformation
ts_wind_data <- ts(df_clean[,3],frequency=12)

#decomposition
decompose_wind_data=decompose(ts_wind_data, type="additive")
plot(decompose_wind_data)
```

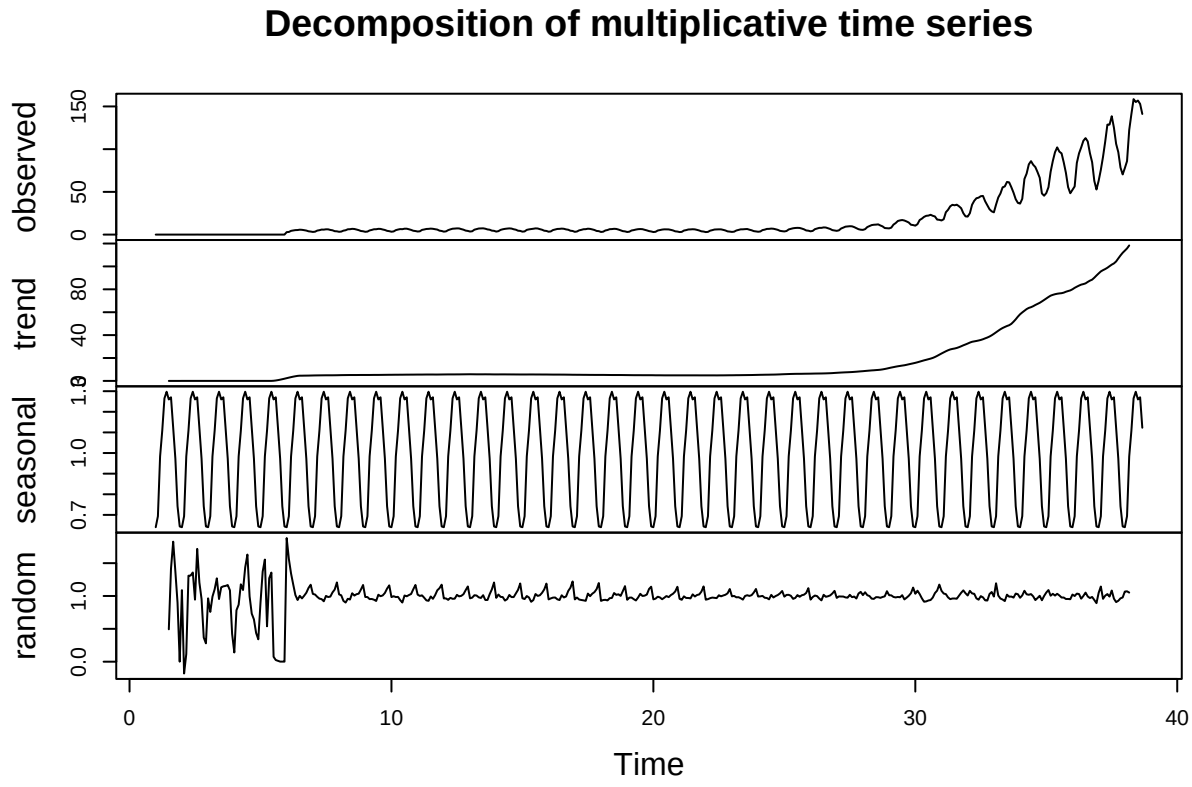


Observations: The trend component identifies a clearly increasing trend of wind energy consumption all the way up until it flattens a bit at the end (possibly due to COVID-19 as the last data point is from September 2021). The random component shows minor variability throughout the dataset until the end of the series where there is an increase in variability. The random component does not look very random and appears to have its own seasonal trend until it hits the portion of the series with increased variability.

Q4

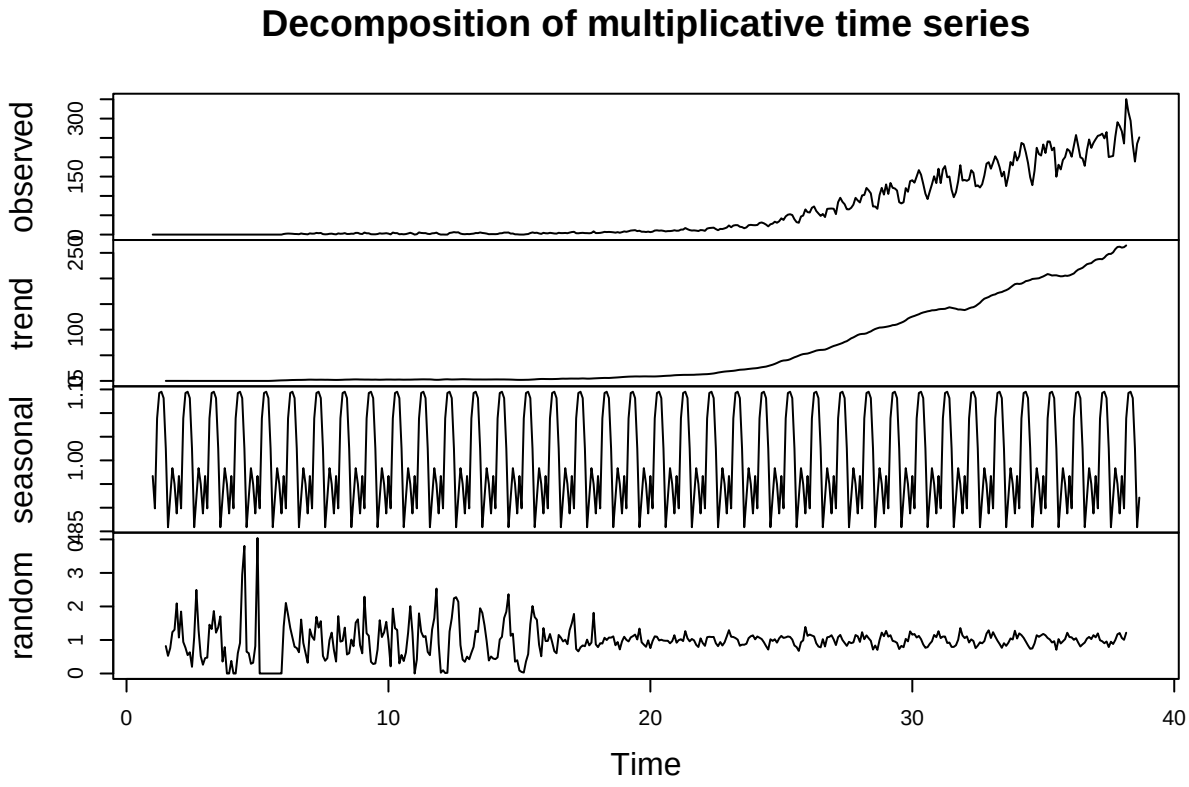
Use the decompose function again but now change the type of the seasonal component from additive to multiplicative. What happened to the random component this time?

```
#solar series

#decomposition
decompose_solar_data_m=decompose(ts_solar_data, type="multiplicative")
plot(decompose_solar_data_m)
```



Observations: The random component of the decomposed solar energy time series now shows high variability at the beginning of the series with the rest of the series tapering off to low variability. It also appears to have a much less significant seasonal trend than the additive time series.

```
#wind series

#decomposition
decompose_wind_data_m=decompose(ts_wind_data, type="multiplicative")
plot(decompose_wind_data_m)
```



Observations: The random component of the decomposed wind energy time series now shows high variability for the first half of the time series with it tapering off to low variabiilty a little before the halfway mark of the series. It also appears to have a much less significant seasonal trend than the additive time series.

Q5

When fitting a model to this data, do you think you need all the historical data? Think about the data from 90s and early 20s. Are there any information from those years we might need to forecast the next six months of Solar and/or Wind consumption. Explain your response.

> Answer: No, all historical data is not needed for this data as there was a significant increase in consumption for both energy sources, starting in the 2010s for solar and mid 2000s for wind. Wind appears to have been affected by the pandemic more than solar, so to forecast the next 6 months, I'd chose data from 2009 to 2019. For solar, since it did not start picking up until the mid 2010s, I would pick data from the beginning of 2015 to the end of 2019 to predict the next six months (and possibly to the end of the dataset as a second prediction to see if they are far off from each other), ignoring the implications of current political events.

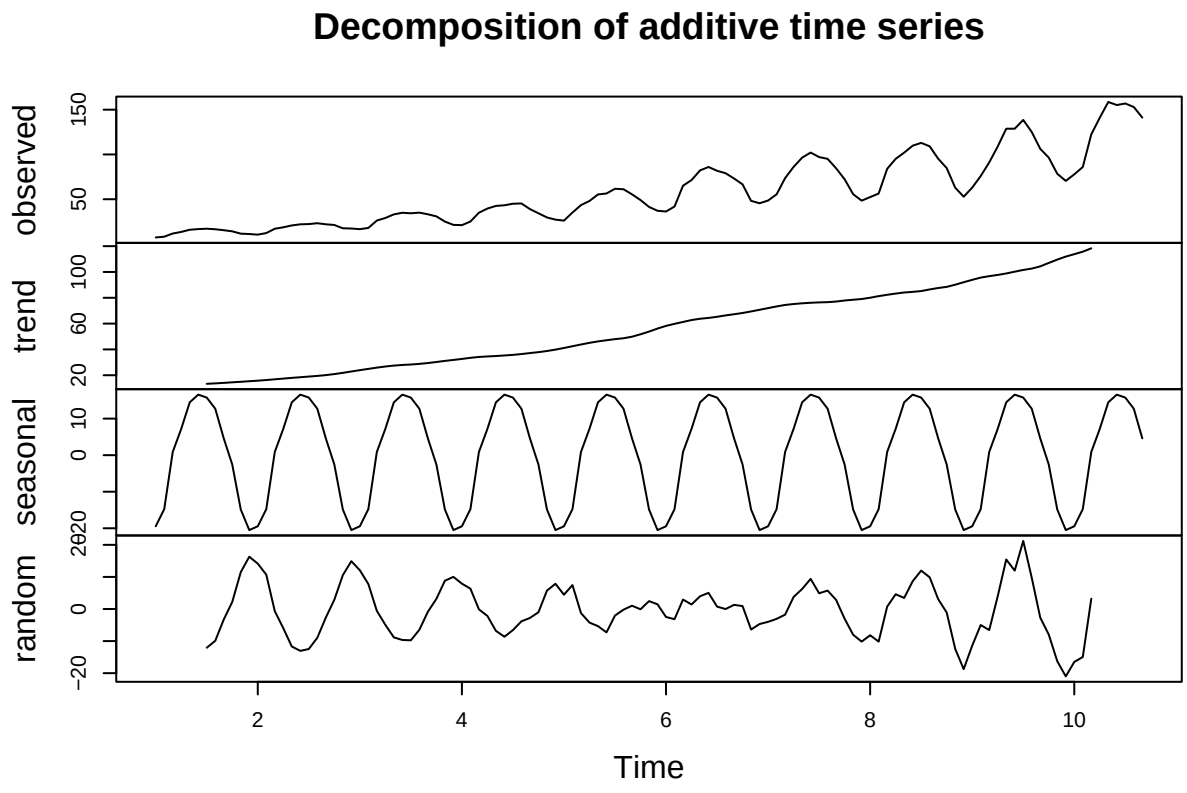
Q6

Create a new time series object where historical data starts on January 2012. Hint: use `filter()` function so that you don't need to point to row numbers, .i.e, `filter(xxxx, year(Date) >= 2012 )`. Apply the decompose function `type=additive` to this new time series. Comment on the results. Does the random component look random? Think about our discussion in class about trying to remove the seasonal component and the challenge of trend on the seasonal component.

```r
#filtering the data frame
df_2012 <- filter(df_clean, year(Month) >= 2012)
head(df_2012)
```

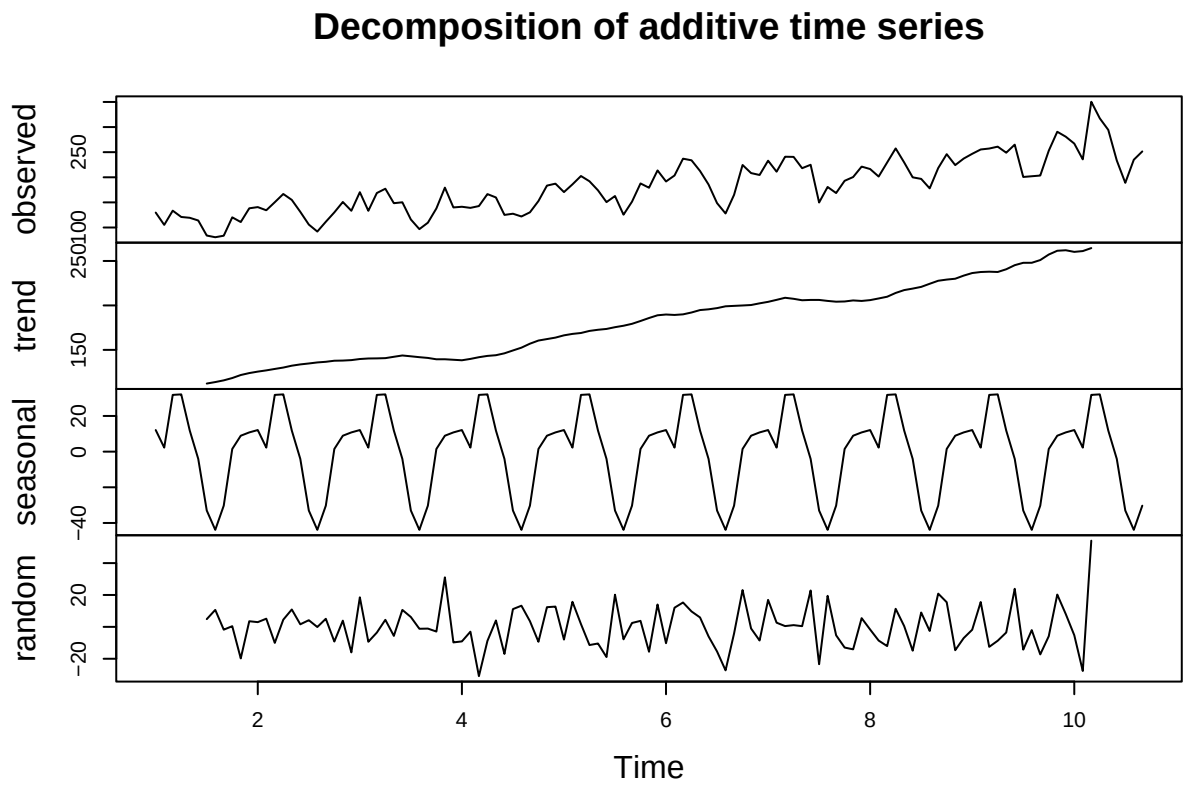
```
##        Month Solar Energy Consumption Wind Energy Consumption
## 1 2012-01-01                    7.288                  129.726
## 2 2012-02-01                    8.165                  105.171
## 3 2012-03-01                   11.678                  133.476
## 4 2012-04-01                   13.478                  120.941
## 5 2012-05-01                   15.933                  119.336
## 6 2012-06-01                   16.651                  113.928
```

```r
#Creating new time series for solar
ts_solar_2012 <- ts(df_2012[,2],frequency=12)

#decomposition of filtered solar data
decompose_solar_2012=decompose(ts_solar_2012, type="additive")
plot(decompose_solar_2012)
```



Observations: The random component for solar energy still appears to have some seasonality, especially at the beginning and end of the time series. This indicates that it is difficult to remove seasonal components from a dataset and that it will take multiple tries and methods in order to get as close to removing seasonality from a dataset as possible.

```r
#Creating new time series for wind
ts_wind_2012 <- ts(df_2012[,3],frequency=12)

#decomposition of filtered wind data
decompose_wind_2012=decompose(ts_wind_2012, type="additive")
plot(decompose_wind_2012)
```



Observations: The random component for wind energy appears to have a less obvious seasonality than the solar data, but still has a bit of seasonality, particularly in the middle 75% of the time series. This further emphasizes that it is difficult to remove seasonal components from a dataset.